An efficient algorithm for dexterous manipulation planning

Anis Sahbani[1], Jean-Philippe Saut[2] and Véronique Perdereau[1]
Université Pierre et Marie Curie - Paris 6, EA 2385.
BP: 252 - 3, rue Galilée, 94200 Ivry-sur-seine, France.
e-mail: [1] sahbani,vperd@ccr.jussieu.fr, [2] jean-philippe.saut@lisif.jussieu.fr

***Abstract*— This paper addresses the dexterous manipulation planning problem of $3D$ rigid objects by a multi-fingered hand. We present a general motion planning algorithm capable to automatically generate specific stable grasps allowing a multi-fingered hand to manipulate rigid objects. It is also capable to address continuous sets of stable grasps, rather than sampling one generally assumed by the previous planners. The algorithm relies on a topological property that characterizes the existence of solutions in the subspace of configurations where the hand grasps the object with four fingers [13]. This property leads to reduce the problem by structuring the search-space. Experiments conducted with the planner demonstrate its efficiency to solve complex dexterous manipulation problems.**

***Keywords*— Dexterous manipulation, stable grasps, motion planning, probabilistic roadmap methods.**

I. Introduction

The work in robot grasping has tried to understand and to reproduce human behavior when manipulating objects. We distinguish two research fields: the generation and the modeling of stable grasps and the dexterous manipulation planning. In this paper, we present an efficient planner that automatically generates the specific stable grasps and computes the sequence of re-grasping operations that make the problem solvable. Motion planning in this context appears as a constrained instance of the coordinated motion planning problem. The solution of the dexterous manipulation planning problem consists in a sequence of sub-paths satisfying these motion constraints. Motions of the hand changing the pose of the object with a fixed grasp are called *transfer paths*, and motions of some fingers of the hand to change from stable grasp are called *re-grasping paths*.
Consider the classic dexterous manipulation planning example illustrated in Figure 1. The 4-fingered hand has to rotate the object (the sphere) from its initial configuration to a desired one. Solving this problem requires to automatically produce the sequence of intermediate stable grasps and re-grasping operations. This example demonstrates, that a dexterous manipulation task leads to a complex sequence of motions, including several re-grasping operations with a continuous test on the stability (and feasibility) of the grasps.
We present in next section existing works on dexterous manipulation planning.

II. Related Work

A first problem formulation was proposed by Li and Canny in the late 80's [18] but they did not give any resolution scheme (for a more recent formulation see [10]).
Trinkle and Hunter [27] built a graph whose nodes are qualitative descriptions of grasps. These descriptions list the contacts between elements of the grasped object and the hand, such as vertices or edges. Linking the nodes is done with a planning method working in joint space and the dexterous manipulation problem solution is found when start and goal configurations are linked to the tree. This work is restricted to a manipulation system with low degree of freedom. Montana [20] proposed a full configuration space description of the multi-fingered manipulation kinematics and presented the finger gaiting example for the twirling of a baton. Han and Trinkle [9] proposed a framework for manipulation planning of a sphere with three fingers. A finger needs to be replaced if it is close to its workspace boundary or if it can not ensure a force closure grasp with any of the two others. Rus [23] proposed a full dynamics algorithm called the *finger tracking* algorithm. The main idea of this algorithm is to use two fixed fingers (with respect to the world frame) and a third one that moves to control the reorientation movement.
Cherif and Gupta [3] used the same principle to plan the re-orientation of a convex object. Three fingertips are fixed

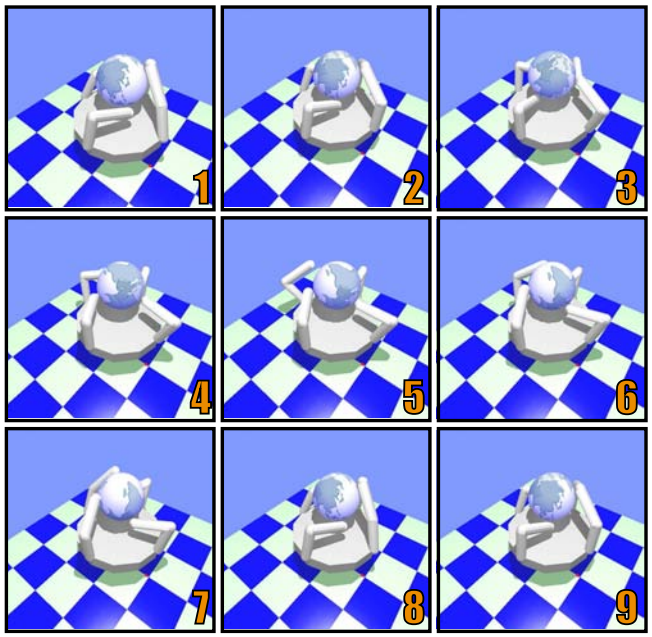

Fig. 1. How to rotate a sphere from its initial position (top,left), to the goal (bottom,left)? The solution requires several stable re-graping operations.

ISBN 978-9973-959-06-5 / SSD © 2007

and the motion of a fourth one is used to rotate the object.
Goodwine [7] and Harmati *et al.* [11] proposed methods based upon nonlinear system control theory. They use motion planning method for smooth system extended to deal with the discontinuities of finger gaiting. The configuration space is divided into strata, each of them corresponding to a particular grasping finger combination. In each stratum the vector fields for the control system are smooth allowing the use of motion planning methods for smooth kinematic nonholonomic systems. Although the problem description by Goodwine can seem close to the one proposed in this paper, the resolution method is quite different.
Sudsang and Phoka [26] proposed a planning method for regrasping with a 4-fingered hand manipulating a polygon. It is based on the construction of a *switching graph*. Each node of this graph is a set of particular grasps called *concurrent grasps*. This technique allows to build a grasping configuration sequence to go from one grasp to another while assuring force-closure property. However, it is only a regrasping planning method and does not regard object motion planning.
Yashima *et al.* [29], [30] proposed a randomized planning architecture based on switching of contact modes. It considers all possible contact modes (sliding, sliding with roll, with spin, *etc.*). Based on RRTs method [17], a global planner builds a random tree to explore the object configuration space and a local planner tries to link the tree nodes. This local planner builds an object trajectory and randomly chooses a contact mode. Then, the inverse kinematic model is used to compute the joint torque trajectories that would lead to the desired object trajectory while satisfying the manipulation constraints.
Xu and Li [28] proposed to use joint space representation of the grasps and to describe the problem as a hybrid automaton which can be seen as a state machine that takes into account both discrete (finger relocation) and continuous (object or finger trajectories) events. They do not present a full resolution method but this is part of their ongoing works.

The contribution in this paper is to propose a more elaborated algorithm, issued from the general approach recently proposed in [13]. Section III recalls notions and briefly explains the method. The algorithm described in section IV computes a graph using probabilistic roadmap techniques [22], [14]. Some dexterous manipulation planning problems solved by the planner are commented in section V.

III. Theoretical Overview

Let $\mathcal{H}$ and $\mathcal{O}$ denote a n-fingered robotic hand and a rigid object in a 3-dimensional workspace. The composite configuration-space of the two systems is $\mathcal{CS} = \mathcal{C}_{hand} \times \mathcal{C}_{obj}$. $\mathcal{CS}_{free}$ is the sub-set in $\mathcal{CS}$ of all admissible configurations (i.e. configurations where the moving bodies do not intersect together or with the obstacles). $\mathcal{GS}_k$ is the sub-space of $\mathcal{CS}_{free}$ defined as the set of free configurations corresponding to all possible grasps of the object $\mathcal{O}$ with k fingers: $\mathcal{GS}_k = \{q \in \mathcal{CS}_{free} \backslash \mathcal{O} \ grasped \ by \ k \ fingers\}$. $\mathcal{GS}_k = \bigcup_{i \in [\![1; C_n^k]\!]} \mathcal{GS}_k^i$ (C_n^k denote a linear combination giving the number of ways to choose k fingers from the n of the robotic hand). Note that $\forall k \in [\![0, \ldots, n-1]\!], \mathcal{GS}_{k+1} \subset \mathcal{GS}_k$. In fact, a $k+1$ fingers grasp is a particular case of k fingers grasp (one of the independent fingers is in a configuration that makes it contact the object surface). The subspace of grasps with exactly k fingers is $\mathcal{GS}_k$ without $\mathcal{GS}_{k+1}$ *i.e.* $\mathcal{GS}_k \backslash \mathcal{GS}_{k+1}$ and is noted $\widetilde{\mathcal{GS}}_k$.
A crucial characteristic of the composite system $(\mathcal{H} + \mathcal{O})$ is the mechanical stability of its configurations. Indeed, some grasps do not allow $\mathcal{H}$ to maintain the grasp of the object. A grasp can be considered as stable if it can exert arbitrary force/torque wrench on $\mathcal{O}$ by applying appropriate contact forces. This is the well-known *force closure* property ([2], [19]). As we assume $\mathcal{O}$ and $\mathcal{H}$ movements to be slow enough to neglect inertial effects, we consider that verifying force closure property at each time is sufficient to guarantee the system stability. Force closure property depends on the contact position and model (point contact with friction, soft contact with elliptic approximation, etc.).
Another important constraint concerns the kinematics of contacts. Indeed, we assume that the contacts between the object and the fingertips can not slide. In the case of a point contact with friction model, a grasp needs three contacts to be stable (at least three fingers participate in the grasp).

The search-space is then reduced to the sub-manifold $\mathcal{GS}_4$ (we can not reduce it to $\mathcal{GS}_3$ because, for a configuration in $\mathcal{GS}_3$, moving one finger to make re-grasping operation makes the grasp unstable). The dexterous manipulation planning problem appears as a constrained path planning problem inside and between the various connected components of $\mathcal{GS}_4$. It is then sufficient (thanks to the reduction property [1]) to study the connectivity of the various components of $\mathcal{GS}_4$ by transfer and re-grasping paths. Note that the connectivity of sub-spaces of $\mathcal{GS}_4$ can be analyzed using motion planning techniques for closed mechanisms.

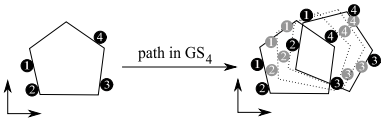

Fig. 2. Illustration of a $\mathcal{GS}_4$ path (in the plane).

The grasp continuity is taken into account by introducing virtual DOFs. These DOFs correspond to continuous contact placement on the object surface. This representation allows to choose configurations respecting the closure of kinematic chains induced by grasping fingers and object as well as to generate path in a $\mathcal{GS}_4^i$. We can thus define a linear path (Fig. 2) between two $\mathcal{GS}_4^i$ configurations. Such a path is obtained by linearly linking the two configuration augmented vectors (augmented with the parameters representing the grasp

continuity).

The idea here is to apply the technique presented in [5] to capture the topology of $\mathcal{GS}_4$ manifold into a probabilistic roadmap, called *manipulation graph* ($\mathcal{MG}$). The connected components of $\mathcal{GS}_4$ are then connected outside this manifold using transfer/re-grasping paths. During a re-grasping path, the object is maintained immobile and some fingers move to change the grasp. A transfer path corresponds to a displacement of the object while the contact positions on object and fingertip surfaces change, only because of the rolling movement between these surfaces. Figure 3 shows an example of such paths for a four-fingered hand. Next section describes the algorithm that we propose to achieve the computation of $\mathcal{MG}$.

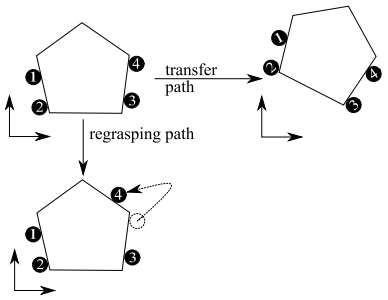

Fig. 3. Illustration of re-grasping and transfer paths for a four-fingered hand (in the plane).

IV. Dexterous Manipulation Planning Algorithm

The algorithm incrementally constructs a dexterous manipulation graph by interleaving two steps: computing $\mathcal{GS}_4$ connected components and linking them by transfer/re-grasping. Like probabilistic roadmap methods [22], [14], the algorithm runs until it exceeds a given number of nodes or a solution for the problem is found (Algorithm 1).

Algorithm 1: Manipulation graph construction

$\mathcal{MG} = \{q_{init}, q_{goal}\}$
$\mathcal{CC} = \{l_1 = \{q_{init}\}, l_2 = \{q_{goal}\}\}$
if $l_1 \equiv l_2$ **then return** *path between* q_{init} *and* q_{goal}
else
  while *(*$l_1 \neq l_2$*)* & *(Node_Nbr* $\leq$ *Nbr_Max)* **do**
    Randomly choose $\alpha \in \{0; 1\}$
    switch α **do**
      case $\alpha = 0$
        Explore_$\mathcal{GS}_4$ (q_{init},q_{goal},$\mathcal{MG}$,$\mathcal{CC}$)
        Node_Nbr ++
      case $\alpha = 1$
        Connect_ Components ($\mathcal{MG}$,$\mathcal{CC}$)
    end
  end
end

A. The α parameter

The choice of α parameter (Algorithm 1) is crucial because it influences greatly the algorithm convergence. Favoring $\alpha = 1$ encourages the discretization of $\widetilde{\mathcal{GS}}_4$ because configurations are added for each transfer/re-grasping connection. However, if $\mathcal{GS}_4$ has numerous connected components, it will improve the convergence speed. It must be tuned according to the problem characteristics (concerning object size or shape complexity and environment obstacles).
The desired behavior of the roadmap builder is to start by constructing portions of the roadmap inside $\mathcal{GS}_4$ components using $\mathcal{GS}_4$ path connections, and then to determine connections of components using transfer/re-grasping paths. Rather than considering separately the two stages, the algorithm uses a more sophisticated way to interleave both phases. The choice of the parameter α is performed by a biased random that depends on the evolution of the size of $\mathcal{MG}$: the first expansion steps start with a low probability to return 1; when the roadmap grows, this probability increases.

B. $\mathcal{GS}_4$ Exploration

The goal of the *"Explore_*$\mathcal{GS}_4$*"* function (Algorithm 1) is to build portions of $\mathcal{MG}$ inside $\mathcal{GS}_4$ in order to capture this subspace topology. Exploring $\mathcal{GS}_4$ in such a way is a motion planning problem for a system containing several closed kinematic loops. One needs to generate configurations verifying chain closures. To solve this problem, we use RLG algorithm [5]. Each chain is divided into an active part and a passive one. The active part configuration is randomly chosen in the accessibility domain of the passive part. The passive part is calculated using inverse geometric models. The grasp stability (force closure property) is checked for every generated configuration along a path. This is done by the function *"Generate_Random_New_Node_*$\mathcal{GS}_4$*"* (Algorithm 2).

Algorithm 2: $\mathcal{GS}_4$ Exploration

N $\leftarrow$ Generate_Random_New_Node_$\mathcal{GS}_4$ ($\mathcal{MG}$)
CC_List $\leftarrow$ {}
Nbr_Connected_CC $= 0$
for $i \leftarrow 1$ **to** $Nbr\_CC(\mathcal{MG})$ **do**
  if *Connected_Inside_*$\mathcal{GS}_4$ *(N,* CC_i*)* **then**
    Nbr_Connected_CC ++
    Add CC_i to CC_List
  end
end
if *(* Nbr_Connected_CC $\neq 0$ *)* **then**
  Merge_CC (CC_List, $\mathcal{MG}$)
  Update_Graph ($\mathcal{MG}$)
end

The function *"Connected_Inside_*$\mathcal{GS}_4$*"* (Algorithm 2) tries to link the nodes of the graph (configurations in $\mathcal{GS}_4$) with linear paths in $\mathcal{GS}_4$. It is thus necessary to be able to connect two configurations in this subspace.

In the works of [24], dealing with robotic arm manipulation, the authors have to represent a grasp continuous change. They simplify the problem using a simple geometry (a parallel jaws gripper grasping a parallelepiped bar). Thus only three DOFs (two in translation and one in rotation) are associated with the grasp. In the dexterous manipulation case, the problem is far more complex because the system has more DOFs and it is desirable to do no reducing assumption on the object shape.
To keep the generality of the approach, it is crucial to use a grasp parameterization allowing continuous changes. Since grasp description needs the contact positions on the object surface, it is necessary to have a parameterization of this surface to be able to compute paths in $\mathcal{GS}_4$. So far, we have supposed object surface to be parameterizable. Actually, most objects do not have such surfaces and not all parameterizations suit our problem. Ideally, contact points must move on object surface along the shortest path and linear variation of parametrized coordinates does not lead to a a shortest path (e.g. spherical coordinates). This problem can be bypassed because actually one just needs to randomly choose points on the object surface and to compute continuous shortest paths on this surface, linking two of these points. A solution is to approximate the surface by a polyhedron. This approximation can be realized with arbitrarily chosen precision. A geodesic computation algorithm (for instance [16]) is used to find the shortest path between two given object surface points. The path is computed as a set of successive segments. The choice of a random contact point is done by first randomly choosing a facet of the polyhedron using a bias on its area, then by choosing a position on this facet.

C. Connections by transfer/re-grasping paths

The function "*connect_components*" (Algorithm 1) tries to link different connected components of the manipulation graph using transfer/re-grasping paths. The transfer path goal is to bring the object configuration from its initial to a final pose. The goal of the re-grasping path is then to bring the hand to a desired grasp. The cited function is detailed in algorithm 3.

Algorithm 3: Connections outside $\mathcal{GS}_4$

```
N ← Node(MG)
Nbr_Connected_CC = 0
for (i ← 1 to Nbr_CC(MG)) & (CC_i ≠ Comp(N)) do
    if Connect_Transfer_Regrasp (N,CC_i) then
        Nbr_Connected_CC ++
    end
end
if ( Nbr_Connected_CC ≠ 0 ) then
    Merge_CC (CC_List, MG)
    Update_Graph (MG)
end
```

- Transfer path computation
  Transfer paths are object movements realized by rolling fingertips on its surface. To compute them, knowing the object trajectory, we need to find finger movements satisfying the constraints of rolling contacts. These constraints are well known (see for instance [4]). Once a contact is known and verifies the appropriate geometric constraints (it must belong to both object and fingertip surfaces, the normal vectors of these two surfaces must have the same direction and the contact point must be reachable by the finger), it must satisfy two kinematic constraints: the relative velocity of object and fingertip at contact point must be null and the contact point velocity must conform with the associate finger kinematic model. Different methods exist to compute trajectories verifying these constraints ([4], [15]). However, they all require a surface parameterization of both object and fingertip. Instead we chose to integrate the constraints numerically. Finger velocities are computed so as to ensure the nullity of the relative velocity of the two contacting bodies, while the two surfaces are constrained to not inter-penetrate (surface distance is computed using polyhedron collision detection techniques). It is only an approximated computation but its precision depends on the integration step size. Reducing the step size increases computation times but, as transfer path computation occurs mainly at the end of the planning algorithm, once a solution is found, it is not a very significant drawback.

- Re-grasping path computation
  To compute the re-grasping paths in a $\widetilde{\mathcal{GS}}_3^i$, a collision free trajectory for the free finger has to be planned. This can be simply done using the RRT method [17].

D. Solving dexterous manipulation queries

Once $\mathcal{MG}$ computed, it can be used for solving several re-orientation problems. This can be performed using the three following steps. First, the start and goal configurations are connected to $\mathcal{MG}$ using paths outside $\mathcal{GS}_4$ (transfer/re-grasping paths) and the graph is searched between the both configurations. $\mathcal{GS}_4$ portions of the solution path are then transformed into a finite sequence of transfer/re-grasping paths. Finally, the solution is smoothed to eliminate unnecessary motions.

V. Experimental Results

The dexterous manipulation planner was implemented within a software currently developed at LISIF [1]. Several environments have been used as test-bed of the planner. In this section, we present the result obtained onto three of them. The computation times correspond to experiments conducted on a PC equipped with a Athlon64 3500+ processor and 1GB memory. The first example, presented in the Figure 1, is the well known sphere reorientation problem. We refer to

[1]Laboratoire des Instruments et Systèmes Ile de France.

it as *Sphere* example. Figure 4 illustrates a problem (*Cubic*) involving a cubic form manipulated object. The last problem (*EllipObst*), shown in Figure 5, consists in the displacement of an ellipsoid in presence of an obstacle. A four fingered hand is used in the three examples. Each finger has three degrees of freedom. The contact model used for force closure test is the *PCWF* (point contact with friction model).

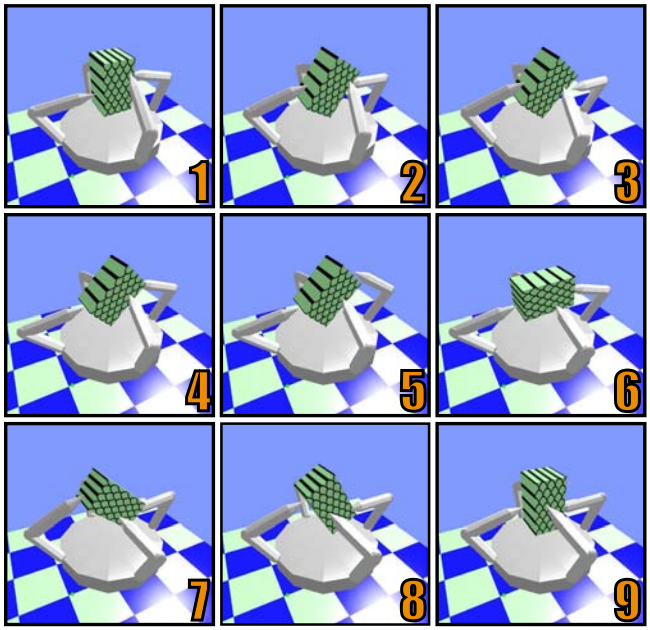

Fig. 4. Manipulation path computed for the *Cubic* problem

In the example *Sphere*, the dexterous manipulation problem is to re-orient the sphere-shaped object, starting from an initial pose to a final one. According to the search space structure in the case of a 4-fingered hand (section III), the Sphere example becomes easy to resolve. $\mathcal{GS}_4$ is composed by a single connected component. But, paths computed in $\mathcal{GS}_4$ are not feasible from the manipulation constraints point of view. Thanks to the reduction property and within the refining process this path is decomposed in transfer and re-grasping paths. This example was solved in less then 20 seconds.

In the *Cubic* example, the task consists in carrying out a full rotation of the object. The main difficulty with this problem lies in the shape of the object. Several re-grasping operations are needed to change grasp from one facet to another. The distribution of the fingers on the object to produce a stable grasp is also more constraining than in the case of the *Sphere* example. Nevertheless, our planner solves this problem in a few seconds.

The *EllipObst* problem is more complex than the two previously presented. The presence of the obstacle increases the number of connected components of $\mathcal{GS}_4$ and also the number of connection tests between the computed nodes in this manifold. Our planner automatically computes the specific jump of the finger presented in Figure 5 (top, right) allowing the re-grasping operation solution of this problem.

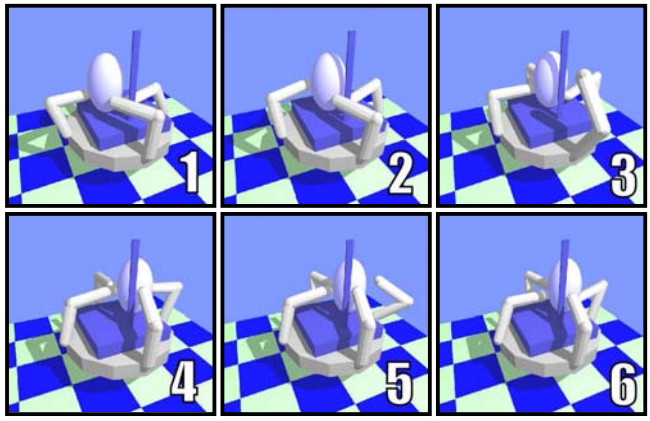

Fig. 5. Manipulation path computed for the *EllipObst* problem.

Table 1 shows for the three examples numerical results of the performance of the algorithm. All examples were solved after less than three minutes of computation. One can also note that the most of computation time is spent for checking the stability and the collision of the generated grasps. This shows the interest of the proposed approach which limits the number of such tests by first computing connected components inside $\mathcal{GS}_4$.

Example	*Sphere*	*Cubic*	*EllipObst*
Average resolution time	17.5 s	27 s	128 s
Average generated node number	196	113	543
Average graph node number	23	27	65

Table 1. Numerical results.

VI. Conclusion

We have presented an efficient algorithm for the dexterous manipulation planning problem. It relies on a structuring of the search-space allowing to directly capture into a probabilistic roadmap the connectivity of the sub-manifolds corresponding to the stable grasps configurations. Due to this structuring, the planner automatically generates, inside continuous domains, a specific sequence of stable grasps and re-grasping operations that make the problem solvable. Experiments conducted with the planner demonstrate its efficiency to solve complex dexterous manipulation problems. There remain several possible improvements, in particular the optimization of the computed paths outside of $\mathcal{GS}_4$ and studying the influence of the parameter α. Also, future work concerns the characterization of the conditions under which we can limit the exploration of the search-space to a restricted manifold, i.e, $\mathcal{GS}_n$.

REFERENCES

[1] R. Alami, J.-P. Laumond, and T. Siméon. Two manipulation planning algorithms. *Algorithmic Foundations of Robotics WAFR94*, AK Peters Publisher, K. Goldberg et al. Editor, 1994.

[2] A. Bicchi. On the closure properties of robotic grasping. *The International Journal of Robotics Research*, 14:319–333, 1995.

[3] M. Cherif and K. Gupta. Planning Quasi-Static Motions for Reconfiguring Objects with a Multi-fingered Robotic Hand. *IEEE Transactions on Robotics and Automation*, 4: 491–503, 1999.

[4] A. Cole, J. Hauser, and S. Sastry. Kinematics and Control of Multifingered Hands With Rolling Contact. *IEEE Transactions on Automatic Control*, 34:398–404, 1989.

[5] J. Cortés, T. Siméon, and J.-P. Laumond. A Random Loop Generator for Planning the Motions of Closed Kinematic Chains using PRM Methods. *IEEE International Conference on Robotics and Automation (ICRA'02)*, 2141–2146, Washington D.C., 2002.

[6] B. Googwine. Stratified Motion Planning with Application to Robotic Finger Gaiting. *Proceedings of the IFAC World Congress*, China, 1998.

[7] B. Goodwine and J. Burdick. Motion Planning for Kinematic Stratified Systems with Application to Quasi-Static Legged Locomotion and Finger Gaiting. *IEEE Transactions on Automatic Control*, 18:209–222, 2002.

[8] S. Gottschalk, M.C. Lin, and D. Manocha. OBBTree: A Hierarchical Structure for Rapid Interference Detection. *Proceedings of ACM Siggraph'96*, 1996.

[9] L. Han, and J.C. Trinkle. Dextrous Manipulation by Rolling and Finger Gaiting *IEEE International Conference on Robotics and Automation (ICRA'98)*, 730–735, 1998.

[10] L. Han, Z. Li, J.C. Trinkle, Z. Qin, and S. Jiang. The Planning and Control of Robot Dextrous Manipulation. *IEEE International Conference on Robotics and Automation (ICRA'98)*, 263–269, 2000.

[11] I. Harmati, B. Lantos, and S. Payandeh. On Fitted Stratified and Semi-Stratified Geometric Manipulation Planning with Fingertip Relocations. *The International Journal of Robotics Research*, 21:489–510, 2002.

[12] P.E. Hart, N.J. Nilsson, and B. Raphael. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Transactions on Systems Science and Cybernetic*, 4:100–107, 1968.

[13] J.-P. Saut, A. Sahbani, and V. Perdereau. A Global Approach for Dexterous Manipulation Planning Using Paths in n-fingers Grasp Subspace. *In the IEEE International Conference on Control, Automation, Robotics and Computer Vision (ICARCV'06)*, Singapore, 2006.

[14] L. Kavraki, and J.-C. Latombe. Randomized Preprocessing of Configuration Space for Fast Path Planning. *IEEE International Conference on Robotics and Automation*, 2138–2139, San Diego, 1994.

[15] B. Kiss, J. Lévine, and B. Lantos. On Motion Planning for Robotic Manipulation with Permanent Rolling Contacts. *The International Journal of Robotics Research*, 21:443–461, 2002.

[16] M. Lanthier, A. Maheshwari, and J.-R. Sack. Approximated Weighted Shortest Paths on Polyhedral Surface. *Proceedings on the 13th Annual ACM Symposium on Computational Geometry*, 274–283, 1997.

[17] S.M. LaValle, and J.J. Kuffner. Randomized kinodynamic planning. *The International Journal of Robotics Research*, 20:378–400, 2001.

[18] Z. Li, J. Canny, and S. Sastry. On Motion Planning for Dexterous Manipulation, Part I: The Problem Formulation. *IEEE Conference on Robotics and Automation (ICRA'89)*, 775–780, 1989.

[19] Y.-H. Liu Qualitative test and force optimization of 3D frictional form-closure grasps using linear programming. *IEEE Transactions on Robotics and Automation*, 15:163–173, 1999.

[20] D.J. Montana The Kinematics of Multi-Fingered Manipulation *IEEE Transactions on Robotics and Automation*, 11:491–503, 1995.

[21] T. Omata, and K. Nagata. Planning Reorientation of an Object with a Multifingered Hand. *IEEE Conference on Robotics and Automation (ICRA'94)*, 4:3104–3110, San Diego, 1994.

[22] M. Overmars, and P. Svestka A Probabilistic learning approach to motion planning. *In Algorithmic Foundations of Robotics (WAFR94)*, AK Peters Publisher, K. Goldberg et al. Editor, 1994.

[23] D. Rus In-hand Dexterous Manipulation of 3D piecewise-smooth objects. *International Journal of Robotics Research*, 1997.

[24] A. Sahbani, T. Siméon, and J. Cortès. A probabilistic algorithm for manipulation planning under continuous grasps and placements. *IEEE International Conference on Intelligent Robots and Systems (IROS'02)*, 1560–1565, 2002.

[25] T. Siméon, J. Cortès, A. Sahbani, and J.-P. Laumond. A General Manipulation Task Planner. *The Algorithmic Foundations of Robotics*, V:311–328, Springer Verlag, 2003.

[26] A. Sudsang, and T. Phoka. Regrasp Planning for a 4-Fingered Hand Manipulating a Polygon. *IEEE International Conference on Robotics and Automation*, 2:2671–2676, 2003.

[27] J.C. Trinklr, and J. Hunter. A Framework For Planning Dexterous Manipulation. *IEEE Conference on Robotics and Automation (ICRA'91)*, 775–780, Sacramento, 1991.

[28] J. Xu, and Z. Li Kinematic Modelling of Multifingered Hand's Finger Gaits as Hybrid Automaton. *The IEEE/RSJ International Conference on Intelligent Robots and Systems*, 3252–3257, 2005.

[29] M. Yashima, and H. Yamaguchi. Dynamic Motion Planning Whole Arm Grasp Systems Based on Switching Contact Modes. *IEEE International Conference on Robotics and Automation*, Washington D.C., 2002.

[30] M. Yashima, Y. Shiina, and H. Yamaguchi. Randomized Manipulation Planning for A Multi-Fingered Hand by Switching Contact Modes. *IEEE International Conference on Robotics and Automation (ICRA'03)*, Taiwan, 2003.

[31] H. Zhang, K Tanie, and H. Maekawa. Dextrous Manipulation Planning by Grasp Transformation. *IEEE Conference on Robotics and Automation (ICRA'96)*, Minneapolis, 1996.